A PATTERN-BASED APPROACH USING COMPOUND UNIT RECOGNITION AND ITS HYBRIDIZATION WITH RULE-BASED TRANSLATION

HANMIN JUNG, SANGHWA YUH, TAEWAN KIM, AND SANGKYU PARK
Electronics and Telecommunications Research Institute, Taejeon, Korea

This paper describes a compound unit (CU) recognizer as a pattern-based approach and its hybridization with rule-based translation. A compound unit is a combined concept including collocations, idioms, and compound nouns. CU recognition reduces part of speech ambiguities by combining several words into a unit and consequently lessening the parsing load. It also provides pretranslated natural equivalents. Our focus in this paper is to obtain flexibility and efficiency from pattern-based machine translation, and high-quality translation by hybridization. A modified trie, our search index structure using "method" strategy is used to manage heterogeneous property of the constituents. Syntactic verification is integrated to obtain precise CU recognition by means of pruning wrongly recognized units that are caused by improper variable hypotheses. The experimental result with verification shows that the precision of CU recognition is increased to 99.69% with 31 CFG rules on the cyclic trie structure for 1,268 Wall Street Journal articles of the Penn Treebank. Another experiment with CU recognition also shows that it raises the understandability of translation for Web documents.

Key words: compound unit (CU), heterogeneous/cyclic trie, pattern-based translation, hybrid translation, syntactic verification, partial parsing.

1. INTRODUCTION

Conventional rule-based machine translation systems often fail to translate certain types of phrases/clauses/sentences. They are collocations (e.g., “They took a walk” and “He made an attempt”), idioms (e.g., “kick the bucket” and “take the bull by the horns”), and compound nouns (e.g., “thrift holding company” and “borrowing authority”). Most of them include fixed expressions that are difficult to treat by conventional syntactic analysis and word-to-word transfer. One means of resolving this problem is to get all fixed expressions from the target domain. This is a prospecting method to make noncompositional problems including the fixed expressions into compositional ones. We define the compound unit (CU) as a combined concept including collocations, idioms, and compound nouns (Jung et al. 1997a; Jung et al. 1997b).

There have been many studies on this topic (Bond, Ogura, and Kawaoka 1995; Lauer and Dras 1994; Li et al. 1995; Yosiyuki, Takenobu, and Hozumi 1994); however, they focus on only a type of the CU categories. Most of them concentrate on compound nouns, which comprise only 56% of all of the CUs we extracted (see Section 6, Experimental Results). Katoh and Aizawa (1995) use compound nouns and collocations, but their patterns are too specific to expand to the real world. Schenk (1986) resolves the idiom recognition problem within a theoretical framework, but he does not consider the syntactic relation between an idiom and its neighbors. This relation is the crucial key to natural translation of a phrase or sentence including an idiom, especially in the case of translating SVO structure (e.g., English) to SOV structure (e.g., Korean and Japanese). None of the studies also shows efficient algorithms to find and process patterns from input sentences. Yoon (1993) uses a simple search algorithm for collocations. However, its matching time is too rapidly increased as the size of collocation entries grows.

CU reduces the number of parts of speech (POS) ambiguities by combining several words
into a unit. It also brings down the loads of morphological/syntactic generation module with predefined natural translations (Lee 1994). Our compound unit recognition described in this paper focuses on achieving flexibility, efficiency, and high-quality translation through hybridization of the pattern-based and rule-based approaches. Our mechanisms feature the following three aspects:

1. the flexibility of pattern matching that has been obtained from variable pattern types (called variable constituents in this paper)
2. a modified trie structure and its search algorithm to manage the heterogeneous property of fixed and variable constituents
3. hybridization of the pattern-based translation and a traditional rule-based translation.

We use a modified trie to control two heterogeneous types (fixed and variable constituents) as an index structure for flexible matching and efficient search. There was no such trie structure with the heterogeneity, only with homogeneity. We also show the matching mechanism for embedded patterns with the use of pseudosyntactic tags.

A CU recognizer cannot use any syntactic information to check syntactic constraints in CUs, but only surface form or its conjugation matching. The result is degradation of recognition reliability (precision). We propose a combination of CU recognizer and syntactic verifier using a partial parsing mechanism. The verification increases the reliability of the recognizer by pruning wrongly recognized CUs. The partial parser operates on a cyclic trie and uses simple CFG rules for fast recognition. Experimental results show that it increases the precision of CU recognition up to as much as 99.69%.

This paper is organized as follows: Section 2 explains several terms concerned with CUs and hybridization, Section 3 describes how to process CU recognition and its syntactic verification, and Section 4 describes the search index and algorithms to treat two heterogeneous types. Section 5 explains the processes and their combination.

2. TERMINOLOGY

1. Fixed and variable constituents. We can extract a CU “keep ~ in ~ mind” from a sentence “I kept the words in my mind yesterday.” “Keep,” “in,” and “mind” always appear in the fixed forms regardless of the context. We define these kinds of words as fixed constituents of CUs. On the other hand, “the words” (#1) and “my” (#2) can be replaced with some other word/phrase/clause (e.g., “your promise on the agreement” (#1) and “his” (#2)) according to the context. We call these variable constituents that should not be omitted. The following example including both fixed and variable constituents shows how they are represented after CU recognition.

[Sentence]
“The operator prevents users from reading the files.” // “prevent #1 from #2”

[After stemming and tagging] // the symbols in parentheses are POS tags
The (DT) operator (NN) prevent (VBZ) user (NNS) from (IN) read (VBG) the (DT) file (NNS) . (. )

[After CU recognition]
The (DT) operator (NN) prevent (VBZ; Identification = *1; CU_POS = VBZ; Start_End = 3-6; Korean = “#1가 #2하는 것을 방해하다”; Japanese = “
2. Pseudo syntactic tag. This tag represents the syntactic role of variable constituents. Zero or a positive integer number following “#” is used for the identification of the constituents. For example, “#1” is the first variable constituent and “#2” the second one in the preceding CU “keep #1 in #2 mind”. The symbol has the form \{*n | n = 1, 2, 3, \ldots\} for a CU and \{#n | n = 1, 2, 3, \ldots\} for its variable constituents. This description makes it possible to process embedded structures of CUs, such as those in the following example. In the case of using directly descriptive forms like “NP” and “VP,” the embedded structure cannot be expressed only with the forms. Our pseudosyntactic tag also includes its descriptive forms to syntactically check with real text.

// CU 'a lot of’ is embedded in another CU “pour #1 into #2”
“pour a lot of balls into . . .” → “pour (*1) a lot of (*2:*1#1) balls into . . . (*1#2)”
(The words in the box are for *1#1 and “a lot of” is for *2)

3. Verb/adjective/noun type. These types are used as the information to integrate recognized CU and rule-based translation. Each of them has its own subcategorization information. For example, “consist of” is verb type “T1.” “T” means transitive verb and “1” indicates that the verb requires a noun object (Longman 1983; Kim et al. 1992).

[Sentence]
“He distinguished himself in the profession.” // “distinguish #1”

[After tagging and stemming]
He (PRP) distinguish (VBD) himself (PRP) in (IN) the (DT) profession (NN). (.)

[After CU recognition]
He (PRP) distinguish (VBD; Identification = *1; CU.POS = VBD; Start.End = 2-3; Verb_Type = T1; Korean = “둘이 나 태나 다”; Japanese = “頭角を現す”; Chinese = “顯露頭角”;) himself (PRP; Identification = *1#1; String.Constraint = “oneself”; Start.End = 3-3) in (IN) the (DT) profession (NN). (.)

3. SYSTEM STRUCTURE

The CU recognizer adapts a plug-in scheme between the morphological and syntactic analyzer. Its results are added to the morphological analysis results in the form of additional features. CU recognition reduces the search space of syntactic analysis and a portion of POS ambiguities. Figure 1 shows the system structure of the recognizer and its syntactic verifier.

\footnote{We have a simple way to get multilingual translation with our CU. The equivalents of other languages are simply added into a corresponding CU without any additional effort.}
to check syntactic constraints. There are other constraints: the POS constraint and the String Constraint. The following example shows how the POS constraint is used (see also Section 2).

```
// “keep #1 in #2 mind”
“keep the words in my mind” → (“keep (VB; Identification = *1; CU.POS = VBD; Start.End = 1-6; Korean = “#1을 #2의 머리에 넣어 주다”; Japanese = “#1を#2の心に刻む”; Chinese = “把#1装在#2的心” the (DT; Identification = *1#1; Start.End = 2-3) word (NNS) in (IN) my (PRP$; identification = *1#2; POS_Constraint=PRP$; Start.End = 5-5) mind (NN)”)"
```

A variable constituent is able to have one or more syntactic constraints. If the validation check for them does not exist, the precision of CU recognition decreases. We introduce partial syntactic analysis to prevent the degradation (Jung et al. 1998). For example, if there is a check routine, the recognizer would find an incorrect CU “take #1 (NP) to” in “But, it doesn’t take much to get burned.” In the case of an input sentence such as “Some researchers have charged that administration is imposing new ideological tests for top scientific posts,” “charge #1 (NP-clause) for” can be accidentally recognized as a CU. However, the result is unreliable because there is no syntactic verification for “#1.” We use a partial parser as the syntactic verifier to raise the reliability of CU recognition. Its operation sequence is as follows:

1. Load CFG rules.
2. Partially parse the variable constituent in the manner of top-down parsing with each of the syntactic constraints (tags) in a selected CU hypothesis and a given sequence of POS tags from the input sentence.
(3) Determine whether the two are syntactically matched. If the match fails, continue to match the other syntactic constraints.
(4) Return the matching result with the hypothesis to the CU recognizer.

The verifier is also able to partially parse embedded syntactic structures, such as “… that Cray Computer anticipates needing perhaps another $120 million in financing beginning next September,” with right recursion from the right-hand side of each grammar rule. We have a grammar index structure with cyclic trie for the recursive grammar. There are currently three left-hand side nodes and 38 right-hand side nodes for 31 CFG rules. We do not have any constraint except for the sequence of the rule description because the partial parser is mainly focused on syntactically verifying in a fast and simple way. The verifier uses a lookahead mechanism to avoid backtracking.

4. COMPOUND UNIT SEARCH

4.1. Search Index Structure

Our index structure for CU recognition consists of three parts: beginning array, constituent trie, and information array (Figure 2). The beginning array helps to rapidly find the index of the constituent to be matched. Each element of the array has at most the two left characters of a CU. Using the first two characters rather than only one or none empirically reduces the number of the traverse on the index to 20 to 80%. The constituent trie is a modified trie structure with heterogeneous nodes: fixed (e.g., “gain,” “a,” “concession”) and variable constituent (e.g., “#1 (NP)”) nodes (Jung et al. 1997c). Previous tries (Cho 1992; Fredkin, Beranek, and Newman 1960; Knuth 1973; Lee 1994) have no way to traverse on both fixed and variable constituents. Therefore, they can only deal with homogeneous property, which lacks the flexibility needed for pattern matching. We modify an ordinary homogeneous trie into a heterogeneous one, and use a “methods” concept to efficiently handle the heterogeneous properties. Some of the variable constituents have one or more conditions that include syntactic tags (e.g., NP, VP, and NP-clause). The element of the information array has all of the information for a CU (e.g., representative POS tag, translation equivalent(s), and CU key). The array is to get the modularization of the index structure.

4.2. Compound Unit Search Algorithms

The principle of CU search is “most-specific-unit-first” (Yoon 1994; Jung et al. 1997c), which means (1) fixed constituent first, variable constituent next, (2) variable constituent with conditions first, variable constituent without condition next, and (3) longer unit first. The longest and most constrained unit for a beginning word is expected to be the best hypothesis for the CU of the word. It implies that at most a CU can be selected for each word; that is, if the word number of the input sentence is $n$, then the number of the found CUs is equal to or less than $n$. We provide three priorities for the traverse to apply our heterogeneous trie structure (Table 1). The other search strategies on the structure are the same as those of other ordinary trie structures. The search principle, which is reflected in the course of constructing a search index, is defined by the “more-specific-than” relation $\gg$ (A and B denote CUs; $A_i$, $B_i$ $(1 \leq i)$ indicate the constituents of A and B).
Figure 2. The search index structure for CU recognition.

Table 1. The Traverse Order on Our Heterogeneous Trie Structure.

<table>
<thead>
<tr>
<th>Priority</th>
<th>Node type</th>
<th>Internal traverse order</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fixed constituents</td>
<td>Alphabetic ascending order, longer length first</td>
</tr>
<tr>
<td>2</td>
<td>Variable constituents with conditions</td>
<td>More conditions first</td>
</tr>
<tr>
<td>3</td>
<td>Variable constituent without any condition</td>
<td>Only one</td>
</tr>
</tbody>
</table>

Fixed constituent $\gg$ variable constituent
If $A \gg B$ iff $A_1 \gg B_1$
Else if $A_1 = B_1$ then $A \gg B$ iff $A_2 A_3 \ldots A_n \gg B_2 B_3 \ldots B_n$
Recursively, where $A = A_1 A_2 A_3 \ldots A_n$ and $B = B_1 B_2 B_3 \ldots B_n$.

We introduce “method” to handle the heterogeneous types in the trie and the traverse on it. The “method” mechanism indicates how to traverse on the trie and what is to be matched on the search index. We can find CU hypotheses in input sentences with this “method.” Traditional trie traversing includes only GO-CHILD and GO-SIBLING; however, with only the two operators it is impossible to find CU on our trie with heterogeneous types. For variable constituents, we introduce two additional “method” operators: SKIP-TO-CHILD and SKIP-TO-NEXT-WORD.

The simplified pseudocodes (see Appendix A) show how the “methods” are used to find CUs. Search time in search_trie( ) is only related to the number of traversed nodes. The number is less than $n^*\Sigma Si$, where $n$ is the number of words in a sentence and $Si$ is the number of siblings for the $i$th word.

In the best case, the time complexity of our recognition algorithms is $\sum_{i=1}^n 1 = n = 0(n)$.
In the worst case, it is $\sum_{i=1}^n (\sum_{d=1}^S Sd) = n(\sum_{d=1}^S Sd) < n(n * S)O(n * n)$, where $d$ is the $d$th sibling node level in the constituent index, $Sd$ is the number of siblings for the current word in the $d$th sibling node level, and $S$ is the maximum value of $Sd$ where $1 \leq d \leq n$. 
TABLE 2. “Methods” and Their Operations.

<table>
<thead>
<tr>
<th>“Method”</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GO-CHILD</td>
<td>Go to child node with next word after matching</td>
</tr>
<tr>
<td>GO-SIBLING</td>
<td>Go to sibling node with current word</td>
</tr>
<tr>
<td>SKIP-TO-CHILD</td>
<td>Unconditionally go to child node with next word</td>
</tr>
<tr>
<td>SKIP-TO-NEXT-WORD</td>
<td>Stay current node with next word</td>
</tr>
</tbody>
</table>

5. HYBRIDIZATION WITH RULE-BASED TRANSLATION

Previous machine translation systems that had only the rule-based approach and that applied only to written text had a number of problems that did not seem likely to be solved in the near future (Choi et al. 1994):

1. the processing of noncontinuous idiomatic expressions
2. the reduction of too many ambiguities in English syntactic analysis
3. robust processing for failed or ill-formed sentences
4. the generation of target sentence style.

The problems can be considered as the factors that have influenced the quality of machine translation systems. This chapter describes the pattern and rule hybrid approaches to resolving the problems of the previous machine translation systems in terms of improvement of translation quality.

CU recognition as our pattern-based mechanism can cause an interfacing problem between pattern-based and rule-based translation. Rule-based parsers typically have direct input from a morphological analyzer without any intermediate process. However, CU recognition as a pattern-based approach changes the whole or partial POS sequence of the input sentence by merging and replacing the parts of speech. To overcome this interface mismatch, we provide the following additional features to the information of each CU: (1) representative POS, (2) verb/adjective/noun types, (3) CU key, and (4) generation code. In two cases the representative POS of a CU is a noun. For simple compound nouns such as “portfolio manager” no additional features are needed because the POS sequence of the input sentence is not changed. If the CU is a derived compound noun such as “ceiling (NP) on” or a phrasal verb such as “talk about,” a feature is needed to represent the relation between the verb/adjective/noun and its object(s). This feature is called the verb/adjective/noun type, which has its subcategorization information. The generation code is used for the interface with the generation module. The CU key is the representative meaning of the CU and it operates to extract the most similar features from the existing equivalent dictionary during English-Korean transfer without any additional word information. For example, the CU “pay on #1 debt” can be captured by its CU key “pay.”

After the syntactic verification for a variable constituent, parsing and transfer/generation should manage the CU information to produce the proper translation result. The following two operations show how to combine pattern-based translation with rule-based translation. First, top-down parsing of syntactic analysis is conducted to obtain an equivalent of the constituent. Second, the result is inserted into the equivalent of its parent CU (Figure 3). The following is an example of the hybrid operation sequence to treat the phrase “pay on real old debt.”
**Figure 3.** The hybridization between pattern-based translation (left box) and rule-based translation (right box).

[Phase 1: after CU recognition] CU recognized verb phrase “pay on... (#1) debt”

(\(\text{VP (}(g\_\text{level 0})(\text{SMHEAD PAY-ON-#1-DEBT})(\text{TENSE PAST})(\text{FFORM PAST EN})\ (V\_\text{TYPE I0}))\ (\text{VERB (}(g\_\text{level 0})(V\_\text{TYPE I0 I3 T1})(\text{ROOT PAY-ON-#1-DEBT})(\text{FORM PAST EN})\ (\text{VERB (}(\text{CU T})(\text{CU_KEY PAY}))\ (\text{LEX PAID-ON-REAL-OLD-DEBT})\ (\text{DICTYPE 1})\ (\text{SEM_MK2 CA})(\text{TOKEN_NO 5})(\text{FORM PAST EN})(\text{ROOT PAY-ON-#1-DEBT})(\text{CAT VERB})(\text{UID 3 2})(\text{LDOCE_TYPE D1 T1 V3 I0 I3 T1})(\text{SITU ACTIV})(\text{VOICE A B})(\text{S\_CASE1 SUBJ})(\text{S\_CASE2 COMP DOBJ})(\text{D\_CASE1 AGT})(\text{D\_CASE2 MGO REC})(\text{C\_LEX2 FOR})(\text{S\_FORM2 13})(\text{SEM_MK1 CA CAH})(\text{OBLIG1 1})(\text{LEX PAID-ON-REAL-OLD-DEBT})\ (\text{#1 #1_PARSE}(\text{CU\_K\_LEX1 #1 2})))))

[Phase 2: after partial parsing of variable constituent “real old”] Partial parsed adverb phrase for #1 “real old”

(\(\text{AP (}(g\_\text{level 0})(A\_\text{TYPE B E}))\ (\text{AP (}(g\_\text{level 0})(A\_\text{TYPE B}))\ (\text{ADJ (}(g\_\text{level 0})(A\_\text{TYPE B}))\ (\text{ADJ (}(\text{LEX REAL})(\text{DICTYPE 1})(\text{TOKEN_NO 7})(\text{ROOT REAL})(\text{CAT ADJ})(\text{UID 1})(A\_\text{TYPE B})(\text{SEM_CAT CENT ORDIN})(\text{S\_CASE1 SUBJ})(\text{D\_CASE1 CHD})(\text{S\_FORM1 1})(\text{SEM_MK1 AG AA})(\text{OBLIG1 1})))))\)

(\(\text{ADJ (}(g\_\text{level 0})(A\_\text{TYPE B E}))\ (\text{ADJ (}(\text{LEX OLD})(\text{DICTYPE 1})(\text{TOKEN_NO 8})(\text{ROOT OLD})(\text{CAT ADJ})(\text{UID 2 1})(A\_\text{TYPE B E})(\text{SEM_CAT CENT AGETE})(\text{THE ADJ PEOP})(\text{S\_CASE1 SUBJ})(\text{D\_CASE1 CHD})(\text{SEM_MK1 CAS CI CAH})(\text{OBLIG1 1})))))\)
6. EXPERIMENTAL RESULTS

Our training corpus is the Wall Street Journal, 1,268 sentences of Penn Treebank (Marcus, Santorini, and Marcinkiewicz 1993). Linguists manually extracted 1,194 CUs from the corpus. Approximately 13,800 CUs that are almost compound nouns and adjectives are automatically extracted from Web pages by the POS pattern finder (n-gram POS matching) and are simply added to the CU dictionary. The average number of words in a sentence is 15.33. Representative parts of speech for CUs include verbs, nouns, prepositions, adverbs, adjectives, and sentences (e.g., It is ~ that ~). Of 1,636 recognized CUs, 56.26% are compound nouns and 29.58% are collocations and phrasal verbs for the training corpus. For the first manually extracted CUs, recall is 97.65% and precision is 98.52% in the same corpus. The default target language is Korean. We use “FromTo/EKTm” for English-Korean translation (Jung et al. 1997d; Sim et al. 1997) and “LogoVista E to J” as the assistance of English-Japanese translation. Following is the sample output from an input with the CU “keep #1(NP) in mind.”

[English] “I kept the promise with him in mind.” // “keep #1 (NP) in mind”
[Korean] “나는 그의 약속을 마음에 새겼다.”
[Japanese] “私は 彼との約束を心に刻んだ。”

The combination of CU recognizer and syntactic verifier increases precision from 98.52% to 99.69%. The remaining gap of 0.31% that is not pruned has wrongly tagged POS words such as “be expected to #1 (verb/noun)” and “sell (verb/noun) #1 from.” The following are the results as syntactically filtered by our verification:

1. “Open #1 (NP) to” in “Sales in stores open more than one year rose 3% to $29.3 million.”
2. “Increase #1 (NP) by” in “Analysts attributed the increases partly to the $4 billion disaster assistance package enacted by Congress.”
3. “Point #1 (NP) at” in “The FT 30-share index settled 16.7 points higher at 1738.1.”

Figure 4 shows a part of the linearity for CU recognition. The average Corr(“method,” word number) is 0.867. Corr(X, Y) is a population characteristic or parameter to measure

---

2We manually evaluated for the two values from the 1,268 sentences.
how strongly X and Y are related in the population (Devore 1990). It is a measure of the degree of linear relationship between X and Y. Absolute value 1 indicates that the relationship is completely linear; 0.867 means the increase of the word number of a sentence is strongly related to the number of used “methods” in linear-like relation. The time complexity of our search algorithms is theoretically $O(n)$ in the best case and less than $O(n^2)$ in the worst case, where $n$ is word number in a sentence. Corr(CUs, trie node) is 0.999. It means that the relation between the number of CUs and trie nodes is completely linear. We also experimentally find that the number of trie nodes and the number of traversed nodes are tightly coupled in the relation.

The time complexity of the syntactic verification using our cyclic trie is also linearly bound by the following experiment. Due to the small size of CFG rules and trie nodes, the verifier imposes little burden on the CU recognizer. Table 3 shows the linear relation between the number of the CU and its recognition time. CU recognition with syntactic verification has little effect on the processing time when compared with the case of recognition without syntactic verification.

We also conducted some experiments on our hybrid approach by using CU recognition. We use understandability (Choi and Kim 1996) as a measure of translation quality. Table 4 shows the degree of the understandability and Table 5 the results of translating some Web homepages. We regard degrees 2, 3, and 4 as the criterion of acceptable translation. The results show that our hybrid mechanism increases the ratio of acceptable translations to 86.8%. It implies that the pretranslated natural translation of CUs helps us to understand target sentences.

7. CONCLUSION

The combination pattern-based approach of using CU recognition with rule-based translation makes the rule-based translator more tractable and adaptable for the open domains on the WWW that have various sentence types. The pattern-based module finds all pretranslated patterns and their information during the morphological and syntactic analysis. Compound
TABLE 3. The Processing Time of CU Recognition with Options under Sun 4/75 that is Similar to the 486-66 MHz PC in Performance.

<table>
<thead>
<tr>
<th>CU</th>
<th>Option</th>
<th>Real time (sec)</th>
<th>User time (sec)</th>
<th>System time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2138</td>
<td>With SV and DL</td>
<td>7.1</td>
<td>0.8</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>With SV and IL</td>
<td>1.3</td>
<td>0.3</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>Without SV, with IL</td>
<td>1.2</td>
<td>0.3</td>
<td>0.7</td>
</tr>
<tr>
<td>3623</td>
<td>With SV and DL</td>
<td>11.5</td>
<td>1.9</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>With SV and IL</td>
<td>1.4</td>
<td>0.3</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>Without SV, with IL</td>
<td>1.3</td>
<td>0.3</td>
<td>0.7</td>
</tr>
<tr>
<td>5026</td>
<td>With SV and DL</td>
<td>15.5</td>
<td>5.4</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>With SV and IL</td>
<td>1.6</td>
<td>0.3</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>Without SV, with IL</td>
<td>1.6</td>
<td>0.3</td>
<td>0.7</td>
</tr>
<tr>
<td>6470</td>
<td>With SV and DL</td>
<td>19.9</td>
<td>10.0</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>With SV and IL</td>
<td>2.0</td>
<td>0.4</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>Without SV, with IL</td>
<td>2.0</td>
<td>0.4</td>
<td>0.7</td>
</tr>
<tr>
<td>7886</td>
<td>With SV and DL</td>
<td>24.7</td>
<td>14.2</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>With SV and IL</td>
<td>2.2</td>
<td>0.4</td>
<td>0.7</td>
</tr>
<tr>
<td></td>
<td>Without SV, with IL</td>
<td>2.1</td>
<td>0.3</td>
<td>0.7</td>
</tr>
</tbody>
</table>

SV: combining with the syntactic verifier; DL: loading from the CU dictionary; IL: loading from the search index.

TABLE 4. The Degree of Understandability.

<table>
<thead>
<tr>
<th>Degree</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 (Perfect)</td>
<td>The meaning of the sentence is perfectly clear.</td>
</tr>
<tr>
<td>3 (Good)</td>
<td>The meaning of the sentence is almost clear.</td>
</tr>
<tr>
<td>2 (OK)</td>
<td>The meaning of the sentence can be understood after several trials.</td>
</tr>
<tr>
<td>1 (Poor)</td>
<td>The meaning of the sentence can be guessed only after a lot of trials.</td>
</tr>
<tr>
<td>0 (Fail)</td>
<td>The meaning of the sentence cannot be guessed at all.</td>
</tr>
</tbody>
</table>

TABLE 5. Translation Results of Five Microsoft Homepages.

<table>
<thead>
<tr>
<th>Degree of understandability</th>
<th>Before hybrid</th>
<th>After hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 (Perfect)</td>
<td>37 (13.5%)</td>
<td>37 (13.6%)</td>
</tr>
<tr>
<td>3 (Good)</td>
<td>45 (16.5%)</td>
<td>47 (17.2%)</td>
</tr>
<tr>
<td>2 (OK)</td>
<td>148 (54.2%) → 84.2%</td>
<td>153 (56.0%) → 86.8%</td>
</tr>
<tr>
<td>1 (Poor)</td>
<td>24 (8.8%)</td>
<td>17 (6.2%)</td>
</tr>
<tr>
<td>0 (Fail)</td>
<td>19 (7.0%)</td>
<td>19 (7.0%)</td>
</tr>
<tr>
<td>Total number of sentences</td>
<td>273 (100%)</td>
<td>273 (100%)</td>
</tr>
</tbody>
</table>

No. of sentences = 273; no. of words = 1,693; average no. of words per sentence = 6.2.
unit recognition described in this paper focuses on obtaining flexibility and efficiency from pattern-based machine translation, and high quality of translation through the hybridization of the pattern-based and rule-based approaches.

The partial parser as a syntactical verifier checks the variable constituents in CUs by the use of simple and right-recursive CFG on our cyclic trie structure. There is a trade-off between processing time and exquisiteness for the recognition. We prefer to build a plug-in module with fast and simple syntactic verification. The strong points of our pattern-based approach are as follows:

1. Pseudosyntactic tags and pretranslated equivalents provide natural translations for target languages.
2. Pseudosyntactic tags for CU and its variable constituents make the processing of embedded CU structure possible.
3. Syntactic tags make the syntactic analyzer parse input sentences with a predictable top-down approach.
4. The modified trie structure using “methods” provides acceptable time complexity against word number in input sentence and CU entries.
5. The time linearity of our search mechanism makes the recognition applicable to practical open domains on the WWW.
6. The combination of CU recognizer and syntactic verifier increases reliability of the recognition.
7. Our hybrid translation using CU recognition increases the understandability of Web documents.

Our future work includes the following efforts: (i) Increase the size of the CUs to obtain the general model for practical domains. We are implementing an automatic CU extractor from the English-Korean bilingual corpus. (ii) Expand multilingual information including equivalents to all the CUs. (iii) Increase verification ability by using additional rule constraints. (iv) Introduce a fail softening mechanism as the complement of our pattern-based approach.

**APPENDIX: PSEUDOCODES OF THE SEARCH_TRIE( ) AND GET_METHOD( )**

```java
search_trie(node, word, skip-or-not) {
    get_method(node, word, skip-or-not);
    switch(method) {
        case GO-CHILD:
            search_trie(child node, next word);
            break;
        case GO-SIBLING:
            search_trie(sibling node, current word);
            break;
        case SKIP-TO-CHILD:
            search_trie(child node, next word, skip);
            break;
        case SKIP-TO-NEXT-WORD:
            search_trie(current node, next word, skip);
            break;
    }
}
```
get
method((node, word, skip-or-not) {   
switch(node) {    
   case fixed constituent:    
      if(matching success)    
         return(GO-CHILD);    
      else if(skip)    
         return(SKIP-TO-NEXT-WORD);    
      else    
         return(GO-SIBLING);    
      break;    
   case variable constituent:    
      if(no condition)    
         return(SKIP-TO-CHILD);    
      else if(matching success)    
         return(GO-CHILD);    
      else    
         return(GO-SIBLING);    
      break;    
}   

REFERENCES


